MS4S09 Coursework 2 - 2020/21

30028762

LATEEF MUIZZ KOLAPO

09/03/2021

Abstract

Forecasting temperature by the use of advance statistical tools is very important in understanding and dealing with the effects of rising or decreasing temperatures. This study uses Time series analysis and predictions, a statistical methods to analyze and forecast temperatures, by using the average max, min and mean temperature for each month of different regions in the United Kingdom measured by the Met Office. The increase in concerns about the effects of rising or decreasing temperature on humans, animals, the climate, oceans, and seasonal patterns provokes the need for accurate models for forecasting the temperatures of the different regions across the globe. This report presents the results of forecasting temperatures using trend analysis, seasonality estimation through seasonal averages and seasonal harmonics, the final model were selected using the autoregressive integrated moving average models

1. DATA COLLECTION

The data used for this study was gotten from the Met Office website where we sourced for the average max, min and mean temperature for each month for 10 different districts in the UK. This gave us 30 different time series to work with, In order to collect the data from this website without having to download all the individual files or read in the files individually, a function was created, which takes two arguments; the region(district) and the temperature(parameter), For each desired time series data, the function reads in the data, transform it using the time series function and outputs a time series data from 1884 to 2020 at a frequency of 12 observations per year which reperesents each individual monthly average in a year. The time series data was saved in a nested lists for each of the different parameters which made the it easy to work with the massive count of this data.

2. Minimum and Maximum evaluation

Two functions were developed to identify the district and date (year and month) of the highest and the lowest max, min and mean temperature. All our time series data are stored in three groups of temperature parameter, which is a nested list a function was created and we employed the lapply function to apply this function across the respective lists within each group of parameters. The output of the implementation of this function shows that for the average monthly minimum temperature;

- The region with the highest temperature is England_SE_and_Central_S , and the date is Aug , 1997
- The region with the lowest temperature is Scotland_E , and the date is Jan , 1895"

The region with the lowest and highest temperature measurement for the average monthly maximum temperature is;

- The region with the highest temperature is East_Anglia , and the date is Jul , 2006
- The region with the lowest temperature is Midlands , and the date is Feb , 1947

Finally, the region with the lowest and highest temperature measurement for the average monthly mean temperature is;

- The region with the highest temperature is East_Anglia , and the date is Jul , 2006
- The region with the lowest temperature is Scotland_E , and the date is Jan , 1895

3 – Exploratory Data Analysis

We performed some exploratory analysis on the time series data and explores some questions about the time series data as seen below.

Which district is the coldest/warmest?

We will be estimating the coldest and warmest region using the following criteria. We have the time series data for the mean daily maximum air temperature, the mean daily minimum and the mean of air for the regions. We will find the coldest region by finding the region with the highest/lowest temperature across these three groups of time series data measured. We observed that the output using this criteria varies among the three different groups of time seris. This can be seen in the output below.

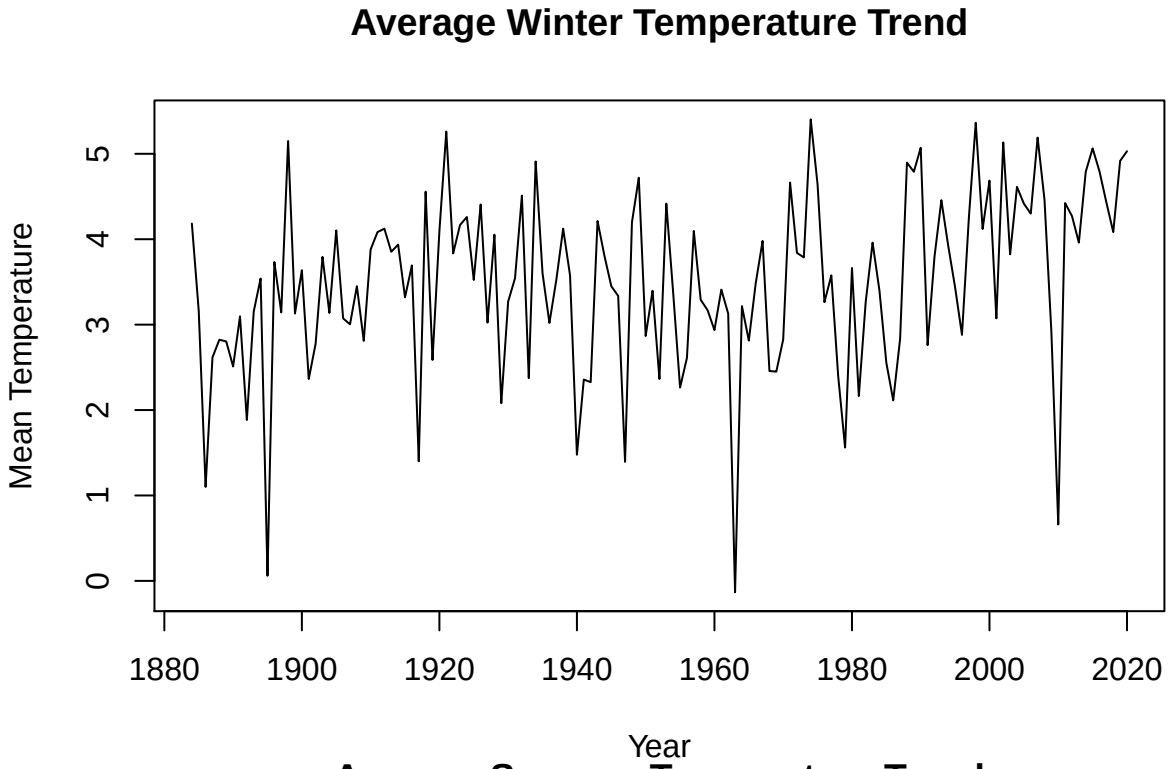
- The region with the highest average monthly mean temperature is England_SE_and_Central_S , While the region with the lowest temperature is Scotland_N
- The region with the highest average monthly minimum temperature is England_SW_and_S_Wales, While the region with the lowest temperature is Scotland_E
- The region with the highest average monthly maximum temperature is England_SE_and_Central_S, While the region with the lowest temperature is Scotland_N

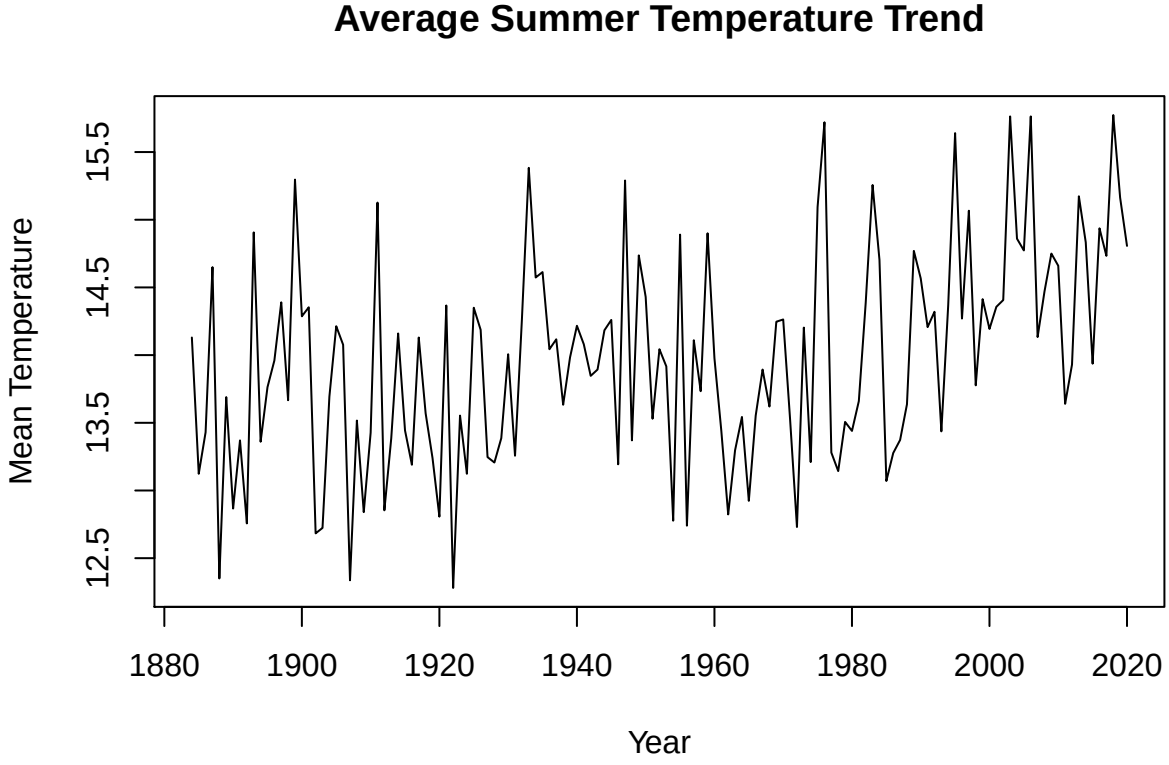
Which district has the widest temperature range?

We created a function that takes returns the highest range for a list of time series data. This was applied to three groups of time series data and we observed that for average monthly mean temperature and average monthly minimum temperature East_Anglia and England_SE_and_Central_S had the widest range but for the average monthly maximum temperature we observed that East Anglia had the widest range of temperatures.

Are winters/summers getting colder/hotter?

We employed data wrangling techniques to group our time series data into two different seasons (winter and summer). We created a function to convert all time series object to data frame of each season(winter and summer). Each row in the dataframe represents a year and the months for a specific season. This function was applied to the average monthly mean temperature, this gives a summary of the occurence for each months in a particular season. A function was then developed to merge all the seasonal dataframe from each region into a single group for a specific season and this was converted to a time series object using the function ts(). The new time series data was then visualized on a time series plot and we observed that the mean temperature for winter months has an upward trend while that of summer months has a downward trend, this means winters are getting hotter while summers are getting colder.





4 – Trend and Seasonality Estimation

We created a function to subset the time series data For each district, and considering the 3 time series: max temp, min temp and mean temp, from 1884 until December 2019. This was implemented using the ts() function which takes start of the series, frequency of the series, and end of the series. We created a function that subsets our time series from 1884 - 2019 called "subset_2019" which was then applied to the entire 30

time series data set. The Lapply function is one used to apply a function to every element in a list, since our 3 groups of time series are stored as a nested list the lapply function was used to apply our subset_2019 function on the 3 different groups of our time series data Tmin, Tmax, Tmean to subset the 30 time series data from 1884-2019. We manually created a time vector for our time series "time.all" which will be used extensively in this analysis.

Compare your results and use appropriate plots and/or tables to confirm your observations.

4.1 Estimating trend

We estimated the trend of each time series using linear, quadratic and cubic regression. A function was developed to apply the 3 different order of polynomial models(linear, quadratic and cubic). The function "run_model" takes a time series data and its time vector and returns its linear, quadratic and cubic models. A function "plot_model" was created which returns a plot of the time series data, linear, quadratic and cubic models all together on a single plot. Finally, We created a function "model_design" which returns the Akaike criterion (AIC) for each model that was passed into its arguments.We are working with data nested into a list and as such we will create a function that can apply the model_design function to a list of time series, this new function was called "apply_model_design". This function "apply_model_design" will return a list of AIC values for the linear, quadratic and cubic models. We then used the apply_model_design on the three groups of time series data we have. The application of this function gives us the AIC value of each region for the different parameters(TMIN,TMEAN and TMAX).

Select a trend model for each time series using an appropriate criteria. Are the models selected all the same? If not is there a pattern depending on the region and/or the group (max, min and mean)?

4.2 Trend Selection

We created a function "which_model" which helps select the best model for a time series, we have the linear, quadratic and cubic AIC values for each region, we will use this function to find the model with the least Akaike criterion(AIC) values which signifies the best model for the time series data. The which_model() function when applied to a list of different time series data returns a dataframe which has a column "Best.Model" which shows the row-wise minimum for each region since each row represents a region and its linear, quadratic and cubic models for a specific parameter. It was observed that all the regions had their best model as the linear model except for two regions(England_E_and_NE and East_Anglia) in the average monthly maximum temperature (Tmax) parameter. The table for best models for each region for a specific parameter can be found below.

Best Model by region for Average monthly minimum temperature

```
##                            Linear Quadratic    Cubic Best.Model
## Northern_Ireland         8750.060  8751.681 8753.238     Linear
## Scotland_N               8809.919  8811.890 8810.050     Linear
## Scotland_E               8965.634  8967.217 8968.253     Linear
## Scotland_W               8845.926  8846.684 8847.663     Linear
## England_E_and_NE         9087.337  9088.720 9088.943     Linear
## England_NW_and_N_Wales   9036.124  9038.017 9037.816     Linear
## Midlands                 9155.355  9156.769 9157.626     Linear
## East_Anglia              9285.648  9287.448 9287.256     Linear
## England_SW_and_S_Wales   8960.401  8962.169 8960.601     Linear
## England_SE_and_Central_S 9221.914  9223.442 9223.818     Linear
```

Best Model by region for Average monthly mean temperature

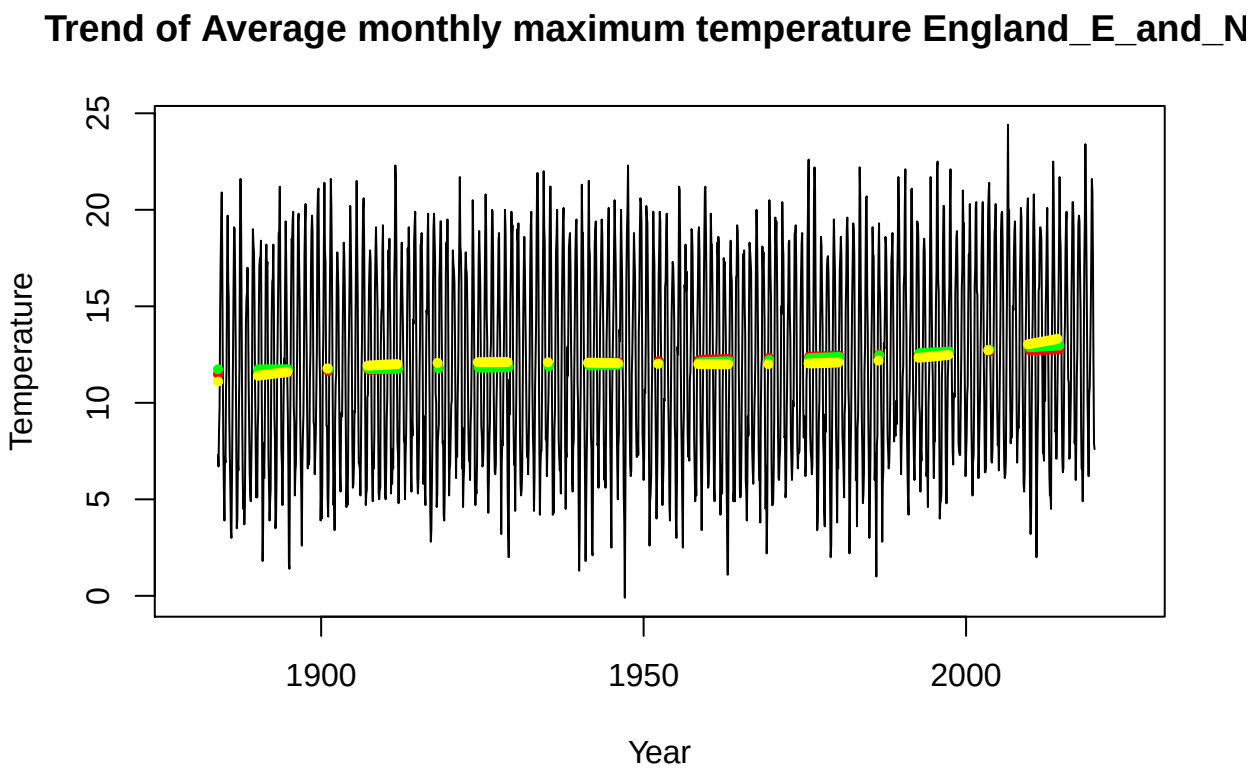
```
##                            Linear Quadratic    Cubic Best.Model
## Northern_Ireland         9053.467  9054.354 9055.851     Linear
## Scotland_N               9062.451  9064.216 9063.682     Linear
## Scotland_E               9366.673  9367.931 9368.846     Linear
## Scotland_W               9174.535  9176.142 9176.027     Linear
```

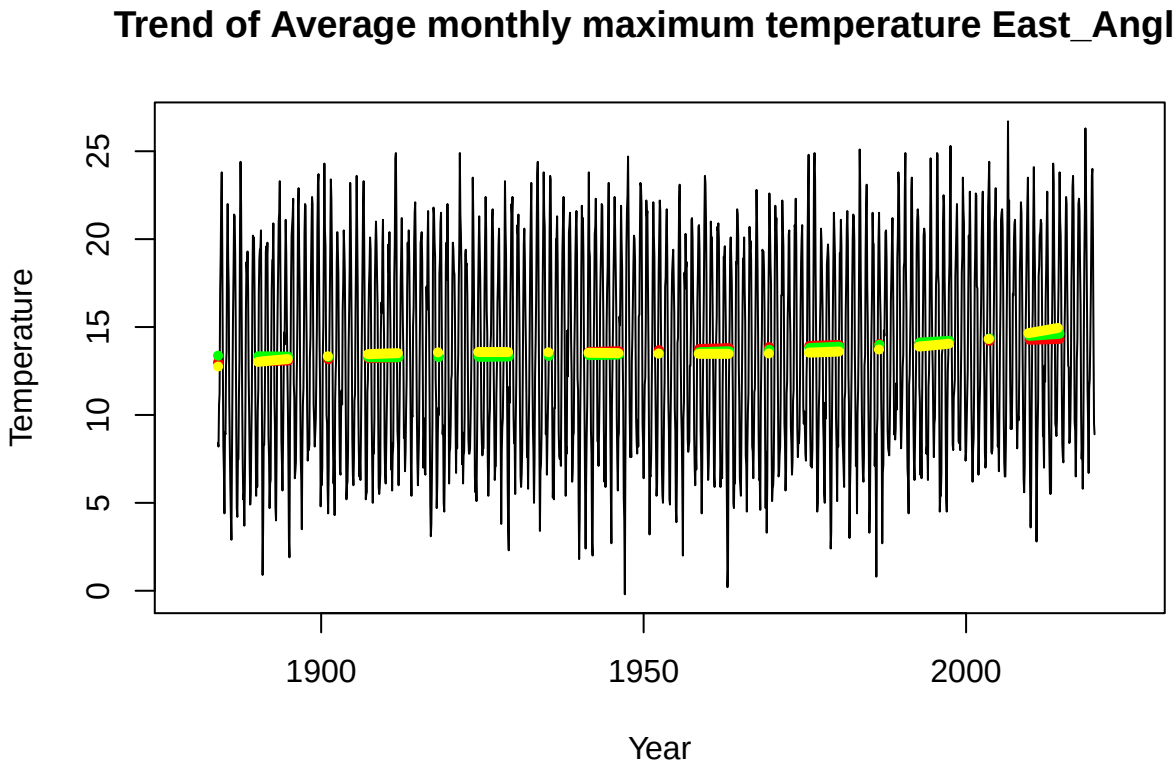
```
## England_E_and_NE          9567.980  9569.231 9568.649     Linear
## England_NW_and_N_Wales    9402.461  9404.136 9403.691     Linear
## Midlands                  9673.179  9673.877 9674.241     Linear
## East_Anglia               9827.018  9828.188 9827.563     Linear
## England_SW_and_S_Wales    9346.494  9347.615 9347.034     Linear
## England_SE_and_Central_S 9728.948  9729.903 9729.898     Linear
```

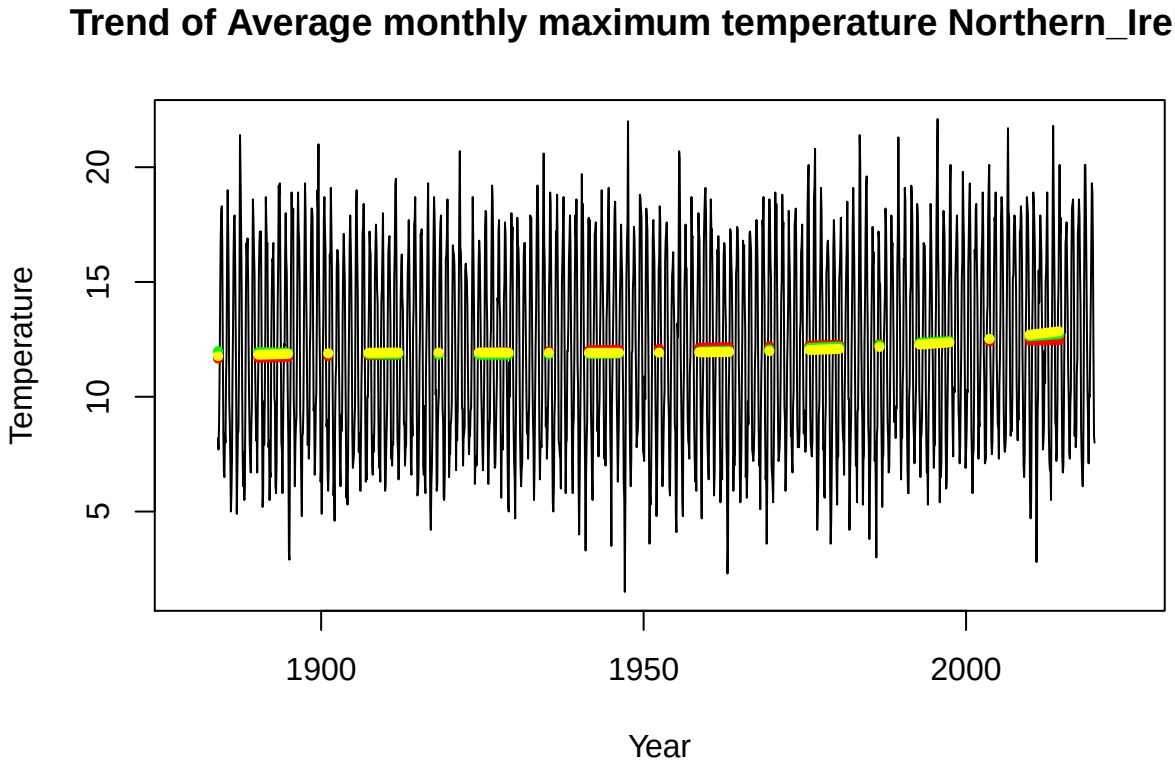
Best Model by region for Average monthly maximum temperature

```
##                            Linear Quadratic     Cubic Best.Model
## Northern_Ireland          9401.134  9401.202  9402.592     Linear
## Scotland_N                9365.093  9366.248  9366.978     Linear
## Scotland_E                9785.996  9786.679  9787.622     Linear
## Scotland_W                9559.381  9561.275  9560.497     Linear
## England_E_and_NE         10025.080 10026.290 10024.750      Cubic
## England_NW_and_N_Wales    9794.982  9796.408  9796.047     Linear
## Midlands                 10170.850 10170.910 10170.870     Linear
## East_Anglia              10329.970 10330.510 10329.770      Cubic
## England_SW_and_S_Wales    9740.923  9741.133  9741.416     Linear
## England_SE_and_Central_S 10205.360 10205.820 10205.500     Linear
```

As stated above we observed that all our model choice for all regions are uniform except England_E_and_NE and East_Anglia for the Tmax parameter, it would be interesting to see a plot of the linear model vs plot of the cubic model. We used the function "plot_model" created above to implement these plots and we observe that there is a difference in the plots for the different regions, while Northern Ireland has a more stable trend, we can see that England_E_and_NE and East_Anglia do have some cubic trend.







4.3 Estimating seasonality using seasonal means and harmonic models

4.3.1 removing Trend using averaging

We will create a function that removes the trend component and returns its monthly average. The function takes two arguments which are the time series data and the polynomial model that bests fit it according to

the Akaike Criterion. All models except England_E_and_NE and East_Anglia for the Tmax parameter have a linear model as their best model. We will split the Tmax time series data into two groups the linear and the cubic groups, this would make it easier to apply functions which are specific to the best model for each region. We will then use the return_month_avg to return the seasonal means of each of the different regions based on their best trend model.

Sample monthly average for Northern Ireland

```
##    Season_Mean Month
## 1   -3.8810267   Jan
## 2   -4.0073597   Feb
## 3   -3.2072220   Mar
## 4   -1.8320843   Apr
## 5    0.5864357   May
## 6    3.3821616   Jun
## 7    5.1492110   Jul
## 8    4.9633192   Aug
## 9    3.2583098   Sep
## 10   0.7672710   Oct
## 11  -1.9127384   Nov
## 12  -3.2662772   Dec
```

4.3.2 Estimate seasonality with seasonal average

The seasonality was estimated using the seasonal means method. We created a function return_seas_avg which takes the time series data and model type. The function models the inputs and returns the seasonal means for each of the time series provided to it. We used lapply to apply the return_seas_avg to the list of time series for the different parameters Tmin, Tmax, Tmean.

4.3.2 Estimating seasonality using harmonic mean We created a function to give the harmonic mean of a time series data. We performed a trend check on the time series and remove the specified trend type(l_model,c_model and q_model). The residuals were derived by subtracting the data from the fitted data. In our dataset since all our best models are linear except for England_E_and_NE and East_Anglia for the average monthly maximum temperature, this means all our models will be linear except these two which will be cubic. In this step we applied the return_harmonic function to all our time series data, a check will be done for models that are significant at a p-value of 0.05.

significant models We created a function to return the significant models for a specific region and harmonic. We need models with a p-value lower than that of the null hypothesis. We create region_harmonics() function to take a list and apply all the function singular_harmonic using lapply() to all the elements in the list provided in the argument. We developed a function grouped_harmonica() that applies the region_harmonics function to a nested list. This would return only the models that has passed the null hypothesis test, hence containing only significant models.

The Null hypothesis and alternative hypothesis are as follows:

- Null hypothesis: The model is not significant
- Alternative hypothesis: The model is significant

Unique models across all time series We created a function that checks the unique models for each element in a list. This unique model would be used to then recreate the harmonic models. We retrieved the best harmonic for each region i.e all harmonics with P-value lesser than the null hypothesis p-value of 0.05. We applied the unique function to all our best model to retrieve only unique models. We observed 5 different unique configurations for our models, this would be used to create 5 different functions, each function will be specific to the unique model configurations found in the implementation below.

create functions to map models

Five different functions were developed inline with the 5 different significant harmonic models we had during the null hypothesis test. The functions are:

- rerun_harmonic1 - [“SIN” “COS”] and [“SIN.1” “SIN.2” “COS.1” “COS.2”]

- rerun_harmonic2 - [“SIN” “COS”], [“SIN.1” “SIN.2” “COS.1” “COS.2”] and [“SIN.1” “SIN.2” “COS.1” “COS.2” “COS.4”]

- rerun_harmonic3 - [“SIN” “COS”], [“SIN.1” “SIN.2” “COS.1” “COS.2”] and [“SIN.1” “SIN.2” “SIN.5” “COS.1” “COS.2”]

- rerun_harmonic4 - [“SIN” “COS”], [“SIN.1” “SIN.2” “COS.1” “COS.2”] and [“SIN.1” “SIN.2” “COS.1” “COS.2” “COS.3”]

- rerun_harmonic5 - [“SIN” “COS”], [“SIN.1” “SIN.2” “COS.1” “COS.2”], [“SIN.1” “SIN.2” “COS.1” “COS.2” “COS.3”] and [“SIN.1” “SIN.2” “SIN.4” “COS.1” “COS.2” “COS.3”]

We observe that all significant models include the first and second harmonics with little variations among the other models. The functions takes the time series data and the best trend model for it l_model, q_model or c_model and returns the harmonic with the lowest Akaike criterion among all the harmonic models implemented within the specific function. Implementing this function gives us the best harmonic model for each of the 30 time series data we are working with.

Subset all the time series into groups of significant harmonic model We have 5 different configurations of significant harmonnic models, we will group each of our time series parameter (Tmin,Tmean and Tmax) to the significant harmonic model group they belong to among these 5 configurations after which their respective function is then applied on each subset. Based on initial analysis we observed that for our average monthly maximum temperature we have two regions with a cubic trend model as their best trend model, we will create a different subset for this group to make the implementation of these functions on the time series data seamless.

4.4 Seasonal model selection Seasonal average or harmonic models?

• Select a seasonal model for each time series using an appropriate criteria. Are the models selected all the same? If not is there a pattern depending on the region and/or the group (max, min and mean)? We now have the different significant harmonic models and the seasonal means of all our time series, we will use the Akaike Criterion to determine the best model for each time series data, as stated above the model with the least AIC will be the best model for the specific time series data.

create a function that selects best model for all different parameters We created a function get_min_AIC that takes 3 arguments the district/region, the best harmonic model, and the seasonal average and returns the model with the lowest Akaike criterion. In this function we combine the AIC of the seasonal average model of the time series and the AIC of the harmonic model and then return the model with the lowest AIC.

ALl TMIN We used the do.call function to re-combine our different configurations for TMIN parameter into one list for each parameter. The get_min_AIC method was then applied on each time series to return the model with the lowest AIC and we observe that seasonal average was not the best for any of the time series in this group.

```
##                           Best Model Tmin
## Northern_Ireland                seas.har2
## Scotland_N                      seas.har2
## Scotland_E                    seas.har3_B
## Scotland_W                      seas.har2
## England_E_and_NE                seas.har2
## England_NW_and_N_Wales          seas.har2
## Midlands                        seas.har2
## East_Anglia                     seas.har2
## England_SW_and_S_Wales        seas.har3_C
## England_SE_and_Central_S        seas.har2
```

ALl TMEAN We used the do.call function to re-combine our different configurations for TMEAN parameter into one list for each parameter. The get_min_AIC method was then applied on each time series to return the model with the lowest AIC and we observe that seasonal average was not the best for any of the time series in this group.

```
##                           Best Models Tmean
## Northern_Ireland                  seas.har2
## Scotland_N                        seas.har2
## Scotland_E                        seas.har2
## Scotland_W                      seas.har3_A
## England_E_and_NE                  seas.har2
## England_NW_and_N_Wales            seas.har2
## Midlands                          seas.har2
## East_Anglia                       seas.har2
## England_SW_and_S_Wales            seas.har2
## England_SE_and_Central_S          seas.har2
```

ALL Tmax We combined the time series with a linear trend into a single list and applied the get_min_AIC and the same was done for time series with a cubic trend.

```
##                           Best Models Tmax
## Northern_Ireland               seas.har3_A
## Scotland_N                     seas.har3_A
## Scotland_E                     seas.har3_A
## Scotland_W                     seas.har3_A
## England_NW_and_N_Wales         seas.har3_A
## Midlands                         seas.har2
## England_SW_and_S_Wales           seas.har4
## England_SE_and_Central_S         seas.har2
## England_E_and_NE                 seas.har2
## East_Anglia                      seas.har2
```

We used the unique function to check the unique best models for each group of our time series and we can see we have 5 different best models distributed among the different time series data. Similar to the approach used above, we will create 5 functions that models our combined trend model and seasonal models for each time series data. **Unique Models**

```
## [1] "seas.har2"   "seas.har3_B" "seas.har3_C" "seas.har3_A" "seas.har4"
```

```
## [1] 5
```

4.5 Building combined model for trend and seasonality

We have 5 different best model distributed among the different groups of time series, we will create 5 functions to implement the combine models based on the groups each time series data belongs to, we will subset the time series into their respective groups and apply these functions across the respective groups. Finally, We combined all the different models for the time series into a variable called "final".

4.6 Creating test set using a combined quadratic and sin-cosine (of order 2) models.

A function return_quad_sin_cos was created to Estimate trend and seasonality using a combined quadratic and sin-cosine (of order 2) models. We created a function to apply the return_quad_sin_cos on a nested list called def_temp. The final outputs were joined into a list and called test.

5 ARMA and Forecasting

5.1 Retrieving the residuals for test and final models

We subsetted the final and test model into subsets of the component parameters (Tmax,Tmean and Tmin), a function was created to derive the residuals for both final and test models. The mapply function was used to dervive the residuals by subtracting the combined model(final and test) from the original time series data. We removed trend and seasonality from each of the 30 time series for both the final and test model and hence we now have 60 residuals time series.

5.2 Fit the residuals with an appropriate ARMA model.

To fit a arma model to all our time series we created a function "fit_fun" which takes a residual as an argument and returns a model for our residuals. This was used to create forecasts for our time series later in this study. The lapply function was used to apply the fit_func function across our nested time series in both final and test set. We assigned the outcome into 6 different lists which reperesents Tmax, Tmean and Tmin for final and test model.

5.3 Forecasting

We will Forecast the average max, min and mean temperature for each month of 2020. We have to forecast the trend, seasonal components and rthe residuals which would then be combined to give our actual forecasts. Earlier in this analysis we observed that our best model for harmonics were splitted into 5 different configurations, we will implement similar solution here creating 5 different functions based on the 5 configurations. The functions were applied to their respective group of time series data and this gave us the final predictions for our models.

-TMIN MODELS

```
## [1] 10
```

-TMEAN MODELS

```
## [1] 10
```

-TMAX MODELS

```
## [1] 10
```

5.4 Model Comparison

We will be evaluating the accuracy of our predictions using the accuracy function of the forecast library, we used the subsetted time series of 2020 data for all our time series as the actual while outcome of our

predictions for 2020 as the predicted. We are going to observe our models performance with unseen data not used when fitting the model. The mapply function was used to apply the model_accuracy function across all our different time series predictions for best test and final.

We observe the rmse for our models below, we can see the rmse for both the test model and final modle, we observed that for some regions the test model performed better than the final model according to the rmse figures in the tables below.

```
##                           Tmin_rmse Tmean_rmse Tmax_rmse
## East_Anglia               1.0137703  1.0591556 1.1416514
## England_E_and_NE          0.7879267  0.8959341 1.1428033
## England_NW_and_N_Wales    0.8558855  0.9830944 1.2888542
## England_SE_and_Central_S 1.0640558  1.1230801 1.3518842
## England_SW_and_S_Wales    0.9544779  0.9894317 1.3759842
## Midlands                  0.9261652  1.0519680 1.3738682
## Northern_Ireland          0.5231145  0.6909344 0.9506735
## Scotland_E                0.8003523  0.8567716 1.0819655
## Scotland_N                0.8038820  0.8478665 0.9273918
## Scotland_W                0.8295469  0.8300845 0.9548110
```

```
##                           Tmin_rmse Tmean_rmse Tmax_rmse
## East_Anglia               1.0057423  1.0363773 1.2934321
## England_E_and_NE          0.7749689  0.8793718 1.1671861
## England_NW_and_N_Wales    0.8349328  0.9472259 1.2555474
## England_SE_and_Central_S 1.0152126  1.0236335 1.3110822
## England_SW_and_S_Wales    0.8702804  0.9193448 1.3035392
## Midlands                  0.8848937  0.9964866 1.3519500
## Northern_Ireland          0.5348322  0.6643862 0.9673044
## Scotland_E                0.7828467  0.8172999 1.0546422
## Scotland_N                0.8094486  0.8560443 0.9894496
## Scotland_W                0.7947144  0.8121647 1.0011004
```

We focused our analysis on the values from the test model predictions, final model predictions and the actual predictions for the South wales and North wales region, a plot was implemented and we observe that the plots are similar for the three different variables for north and south wales.

```
##    test_predictions final_predictions Actual
## 1          1.546477          1.497710    3.2
## 2          1.483202          1.420285    2.6
## 3          2.253805          2.187965    2.0
## 4          3.989963          3.924409    4.4
## 5          6.619674          6.555247    6.2
## 6          9.448657          9.385545   10.2
## 7         11.326805         11.265046   10.8
## 8         11.346451         11.286073   12.2
## 9          9.489493          9.430523    8.6
## 10         6.644874          6.587314    6.5
## 11         3.978881          3.922690    5.2
## 12         2.218433          2.163558    1.9
```

```
##    test_predictions final_predictions Actual
## 1          2.369714          2.247397    3.8
## 2          2.292664          2.106081    4.0
## 3          2.973141          2.908926    3.0
## 4          4.636609          4.400516    5.6
## 5          7.224215          7.157505    6.7
## 6         10.031540          9.841134   10.3
```

```
## 7          11.901634          11.757708   11.2
## 8          11.938664          11.841460   13.1
## 9          10.142091           9.922029    9.5
## 10          7.397123           7.348201    7.5
## 11          4.833470           4.616245    5.6
## 12          3.128227           3.036602    3.1
```